Mars Exploration Rovers Surface Fault Protection

Tracy Neilson
Spacecraft Systems Engineering
Jet Propulsion Laboratory, California Institute of Technology
Pasadena, CA 91109, USA
tracy.a.neilson@jpl.nasa.gov

Abstract - *The Mars Exploration Rovers surface fault protection design was influenced by the need for the solar powered rovers to recharge their batteries during the day to survive the night. The rovers were required to autonomously maintain thermal stability, and initiate reliable communication with orbiting assets or directly to Earth, while maintaining their energy balance. This paper will describe the system fault protection design for the surface phase of the mission, including hardware descriptions and software algorithms. Additionally, a few in-flight experiences are described, including the Spirit FLASH memory anomaly and the Opportunity "stuck-on" heater failure.*

Keywords: System Fault Protection, Mars Exploration Rovers, robotics, spacecraft surface operations

1 Introduction

NASA formally approved the Mars Exploration Rovers (MER) project in July 2000, with less than three years to get to the launch pad. The two rovers, Spirit and Opportunity, launched in 2003, and successfully landed on Mars on January 4 and January 25, 2004, respectively. Over one year later, they continue to explore Mars in extended mission operations.

Fault protection[1] for these rovers was incorporated into every subsystem, built into hardware and software, and system engineered to make sure it all works well together. Fault protection objectives and priorities varied depending on the phase of the mission. During the cruise to Mars, there were no time-critical events, so the system fault protection design was required to put the spin-stabilized spacecraft in a power positive, communicative, and thermally stable state. During Entry, Descent, and Landing (EDL), an extremely time-sensitive period, the fault protection system was designed to use all available resources to ensure a safe landing. For surface rover operations, time was a consumable for the prime 90-sol[2] mission. The fault protection design had to protect the mission science objectives without compromising vehicle health. A quick recovery back to nominal operations in the event of a fault was a primary goal.

2 System Description

The solar powered rovers require a "sleep mode" to recharge the batteries each sol. Sleep mode includes powering off the rover avionics, including the Central Processing Unit (CPU), so the hardware must maintain the safe thermal and power states. Communication requires flight software, so the rover must reliably wake from sleep mode and initiate communication, without intervention from the operations team.

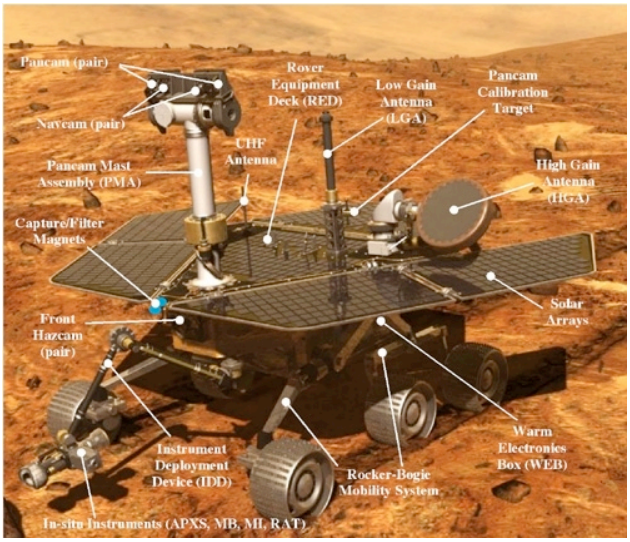

Figure 1: Mars Exploration Rover Detail

2.1 Rover Hardware Overview

Each rover has one Rad-6000 CPU (Rad6k). Onboard memory includes volatile RAM (128 Mbytes) and EDAC-protected non-volatile memory so the system can retain data without power. The non-volatile memory consists of 256 Mbytes of FLASH memory and 11 Mbytes of EEPROM[3].

[1] Per Ref [1], fault protection refers to the treatment of faults in the design and operation of a system. Onboard fault protection, also known as Flight System Fault Detection, Isolation, and Recovery, is the mitigation after faults have caused errors.

[2] A sol is a Martian day, 24 hours 37 minutes long.

[3] There is no EDAC protection on the 3 Mbytes of EEPROM that comes with the Rad6K.

Redundant mechanical thermostats on key components keep the system within flight allowable temperatures. The setpoints on the thermostats are staggered so only one heater is powered on at a time, but the backup heater will still keep the device safe if the primary unit fails. The operations team controls the warm-up heaters for external actuators and camera electronics. To protect against overheating (if a heater switch fails on or if the rover is sleeping), a bimetallic thermostat assembly box may cut off the heater circuit.[4]

Each rover's power system has solar arrays (providing approximately 900 W-hrs per sol at the beginning of the surface mission), and two 8 amp-hr Li-Ion secondary batteries. One Battery Charge Board (BCB) with two identical sides (each dedicated to one battery) provides protection against cell shorts and overdischarge. The power bus is autonomously regulated by the rover shunt limiter, which moves excess energy to the shunt circuits. If the rover power distribution unit detects a low bus voltage, it will power off non-essential loads, including the CPU.

The telecom system includes one Small Deep Space Transponder (SDST), two Solid State Power Amplifiers (SSPAs), one fixed monopole Low Gain Antenna (LGA), one articulated High Gain Antenna (HGA) for tracking the Earth, and one UHF radio for communicating with the orbiting assets.

Time knowledge is maintained with a hardware-based mission clock while the CPU is powered off. The mission clock FPGA is powered directly from the batteries. Co-located on this FPGA is the alarm clock, which is used to trigger the BCB to turn on the avionics.

2.2 Communication Behavior

The rover fault protection design must establish communication autonomously, so the fault protection takes advantage of the flight software communication behavior. This on-board algorithm performs all the actions required to establish and maintain direct-to-Earth (DTE) communication or communication with the orbiters as they pass overhead. The operations team loads several weeks' worth of communication windows onboard the rovers. These windows contain all the information required to perform the communication link, including start time, duration, hardware configuration, and rates. The operations team then designs the operational sequences around these windows, or if necessary, the windows may be modified.

Before the window transmission time, the flight software retrieves data from non-volatile memory and prepares the data for transmission. If requested, the flight software will turn on HGA actuator heaters or perform a new attitude estimate by taking images of the sun. If the window specifies the HGA, the flight software will point the HGA to track the Earth. For either DTE or UHF windows, the coax and waveguide transfer switches are configured, and the X-band power amplifier or UHF is turned on at transmit time. The flight software then feeds packetized telemetry to the radio. At the end of the window, the flight software stows the HGA (if it was the selected antenna) and turns off the transmitter. The receiver stays on while the rover is awake, so the rover remains commandable through the LGA.

2.3 Shutdown and Wakeup Behavior

The backbone of the surface fault protection is the autonomous shutdown and wakeup behavior. This algorithm wakes up the rover when the solar arrays can support the loads required for communication (in receive mode), and it puts the rover back to sleep once the solar arrays alone can no longer support those loads. This strategy maximizes the time for communication with Earth, while still allowing the battery to charge. The amount of energy supplied by the batteries during the night for survival heating is minimized because energy is dissipated inside the rover during the day, and then used as thermal inertia during the night.

Shutdown refers to shutting down the CPU and the avionics. The BCB stays powered, as well as the mission clock and alarm clock. Warm-up heaters and two of the science instruments may also stay on while the rest of the rover sleeps.

During nominal operations, the operations team designs sequences that command shutdowns and include wakeup times to resume the sequence. Upon each commanded shutdown, flight software examines the desired time to resume the sequence and the time of the next communication window, and then sets the alarm clock to the earlier time.

Two triggers may wake up the rovers: Solar wakeup or the alarm clock. The BCB declares the solar wakeup after the solar array current has been greater than 2.0 amps for over 10 minutes (and at least 16 hours have passed since the last solar wakeup). The BCB responds to either wakeup signal by turning on the CPU.

Autonomous shutdown mode is invoked when sequences are inactive or in stasis. In this mode, the flight software will start the shutdown process if there is no communication window active, and either the solar array current is lower than a configurable parameter or the vehicle has been awake too long. If it is time to shut down, the alarm clock is again set such that the vehicle will be ready for the next communication window or for the sequence to resume.

[4] Ref [2] describes the MER thermal design.

Figure 2 describes a typical day in autonomous mode, with alarm clock wakeups for the UHF passes and solar wakeup at 09:00 Local Solar Time (LST). Solar wakeup could occur almost anytime in the morning, depending on the time of year, the atmospheric dust, the dust on the solar panels, and the tilt of the rover. In this scenario, the DTE windows occur while the rover is still awake, but the rover will wake up for any communication window if necessary.

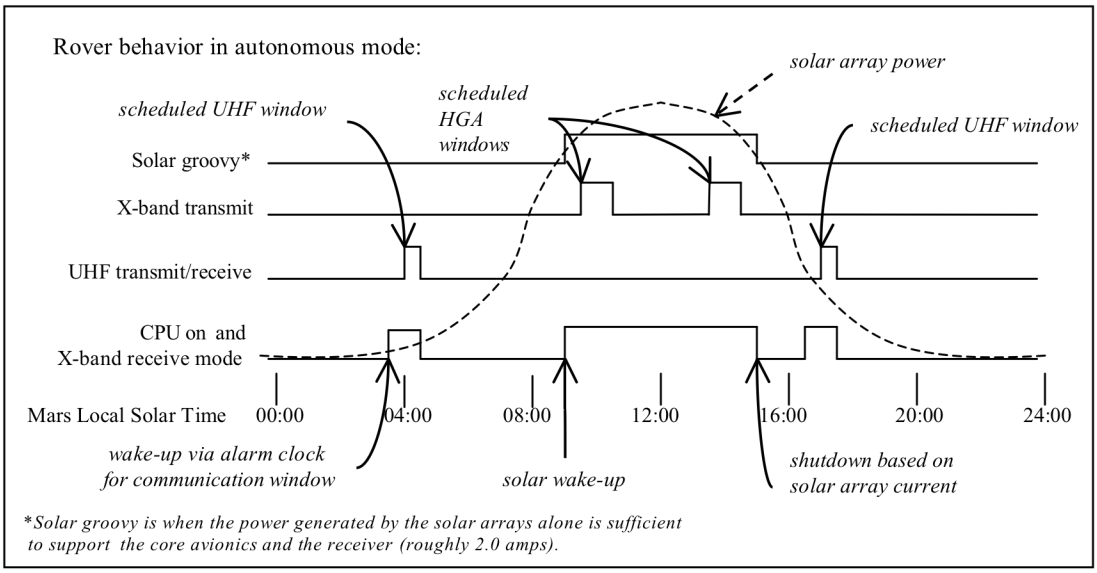

Figure 2: Autonomous shutdown and wakeup scenario

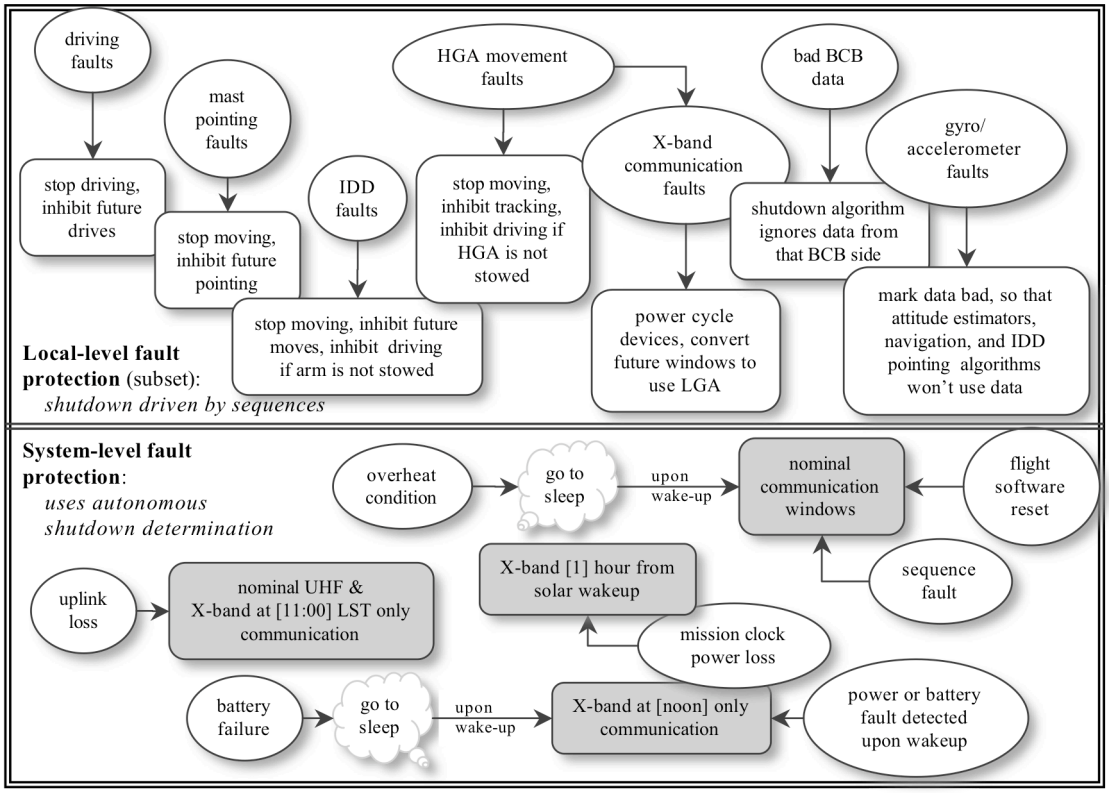

Figure 3: Surface fault protection overview

2.4 Surface Fault Protection Overview

Fault protection on the rovers is handled in a number of ways. This section provides an overview of the rovers' on-board fault protection and gives examples of a few of the responses to faults (refer to Figure 3).

The hardware protection is always available. For example, the BCB FPGA can remove a battery from the bus if it detects a low cell voltage, or the BCB can disable the battery from charging if it detects a cell overcharge.

The flight software may try to establish a hardware state several times if the read-back state doesn't match the predicted state. In other cases, the software may just mark the data "suspect" so other parts of the flight software may take appropriate action. For example, if the flight software detects parity errors in the BCB data, it can mark the data "bad". The autonomous shutdown algorithm then disregards the information from that side of the BCB.

Subsystem level fault protection may declare a function unusable. For example, if a motor overheats while moving the IDD (Instrument Deployment Device or "arm"), the flight software declares an IDD error. Other sequences continue, but commands to move the IDD are rejected. If the arm is not in the stowed configuration, the vehicle safety check will not allow driving, but unrelated activities (such as panorama camera imaging or communication windows) are allowed to continue.

If faults are detected in the X-band telecom system or there are problems pointing the HGA, the fault response will convert the HGA windows to use the LGA with lower data rates, at the same scheduled time.

System fault responses take advantage of the autonomous shutdown/wakeup behavior and the communication behavior to put the rover in a power-safe, thermal-safe, and communicative state. Responses to flight software resets, thermal faults, power faults, and no uplink are examples of responses that use autonomous shutdown and the communication behavior.

Software errors are detected either within the applications or by a software health function that continuously checks for unresponsive or suspended tasks. If severe errors are detected, the flight software will force a reset of the flight computer, causing a flight software initialization. The system response (after initialization) turns off science instruments and warm-up heaters. Survival heaters are reinforced on and all sequences are deactivated. After a flight software reset, the vehicle continues to perform scheduled communication windows in autonomous shutdown mode.

If the flight software detects a severe error during the initialization process, it will delay the reset until a minimum time period has passed. This "delayed reset" is intended to allow the operations team to intervene. If the flight software continues to reset, the system will modify the delayed reset time interval and the boot logic tries alternating copies of the flight software.

The response to low batteries or power faults does not use the scheduled communication windows. If the rover is awake when a low battery situation occurs, the flight software has 60 seconds to quickly shut down before the BCB removes the batteries from the power bus. If it is nighttime when the BCB detects a low battery condition, the BCB takes the battery offline and the power bus crashes. Eventually, when the sun rises, the solar array power supports the bus and the BCB. As the batteries charge back up, the BCB puts them back online to support the bus. At the next solar wakeup, the flight software schedules one LGA communication window at a predetermined hour (11:00 LST) to report to Earth. No UHF windows are attempted because these usually occur in the early morning or late afternoon, when the available solar power is low. The vehicle remains in this configuration (with autonomous shutdown mode active, in receive mode via the LGA, performing one DTE window per day) until the operations team reconfigures the vehicle to resume normal operations.

An uplink loss fault is declared upon the expiration of the uplink loss timer. The operations team sets this timer every sol, with a value roughly equal to three sols in the future. The operations team may debug problems without resetting the timer, but if unsuccessful, the fault response takes over at timer expiration. The response deactivates all sequences to initiate autonomous shutdown mode and schedules an 11:00 LST DTE window. The difference from the low battery response is that the uplink loss response also executes the UHF windows and uses different telecom configurations.

The flight software arbitrates if various fault responses conflict. For example, the system reacts to a low battery event by shutting down immediately, then autonomously performs only one communication window a day. If, after several days, the operations team still hasn't regained control of the vehicle, the uplink loss response will initiate more communication windows, including those with the overhead orbiters.

3 In-flight Experience

The rovers have experienced subsystem level problems with driving, pointing the cameras, operating the science instruments, and placing the IDD. In these cases, the flight software appropriately marked those activities "unusable" and the sequences continued on. The more severe system level fault experiences are described in this section.

3.1 Cruise Solar Flare

Although this event happened during the cruise phase of the mission, the operations team invoked a surface-designed behavior to recover the vehicles. On October 28, 2003, while both rovers were on their way to Mars, a large coronal mass ejection (solar storm) occurred. Star scanners on both Spirit and Opportunity were saturated and 3-axis attitude knowledge was lost. The flight software dropped the attitude determination mode back to a 2-axis view using only the sun sensor.

Another Mars spacecraft, the Odyssey orbiter, detected corrupted RAM due to the same solar storm. Since the MER vehicles do not have a "RAM scrubber" detection mechanism, there was a fear that there might be corrupted areas of RAM which were not accessed during cruise, but would be used during EDL. The only way to clear any solar flare-induced upset and to ensure a clean memory was to power cycle the CPU. Rather than deal with the unknown, the MER Project opted to respond as if there was a problem. This involved manually power cycling the CPU on both vehicles, using a shutdown command, which forced all the hardware and software memory checks to run and map around any damaged RAM. Wakeup was successfully triggered by the alarm clock, the BCB turned on the CPU, and the flight software booted and initialized without incident. No memory corruption was detected on either vehicle. The operations team successfully commanded the spacecraft back to the 3-axis attitude knowledge mode.

3.2 Flight Software Initiated Resets

Both Spirit and Opportunity have suffered flight software-initiated resets on Mars. Only the first events will be addressed in this paper. Spirit's first reset was the FLASH memory anomaly described later and Opportunity's first reset is described here.

During solar conjunction, the operations team was performing a commanding experiment when Opportunity's processor reset. When the rover receives a command, the hardware command decoder (HCD) checks for errors. The HCD corrects all single-bit errors and flags all double-bit errors to the flight software. If three or more errors exist, the HCD Single Error Correct/Double Error Detect code is overwhelmed and it may erroneously 'correct' already corrupted codeblocks or it may not detect any codeblock corruption. In either case, corrupted codeblocks may be passed to flight software without being flagged as such by the hardware. If it is passed a bad code block, the flight software has an error in which the length field from the uplink protocol is not checked for validity before use. The anomaly team concluded that this was the probable cause of the reset.

3.3 Spirit FLASH Memory Anomaly

After 17 sols of successful operations, Spirit's DTE signal suddenly dropped out. Originally, blame was placed on weather at the Deep Space Network tracking station, but the next few communication windows failed to produce any signal. One UHF window produced only pseudo noise, indicating that the vehicle had woken up and turned on the radio. The anomaly team sent commands to Spirit, and it occasionally responded with a carrier-only signal.

Finally, on Sol 21, Spirit sent enough telemetry for the team to deduce that the rover was rebooting over and over and that it was not successfully accessing the FLASH memory. The battery state-of-charge and the thermal telemetry indicated that the rover had not shut down overnight. The operations team commanded an emergency shutdown, but that command failed.

The next sol, the anomaly team sent a command to place the rover in "crippled mode." In this mode, the flight software does not use the FLASH memory file system. This command appeared to fail as well, because the rover would not respond to any commands. An hour later, the rover autonomously initiated DTE communication at 11:00 LST, the fault window time. The batteries had drained overnight, so the BCB had disconnected them from the power bus, and the rover browned out (as expected). During this window, the anomaly team verified that Spirit had stopped rebooting, proving that the crippled mode command had indeed worked. Crippled mode is a volatile configuration, reset each shutdown, so the operations team had to repeat the command each sol to prevent the continuous reboots, until the anomaly team could come up with a fix.

After reconstructing the small amount of telemetry retrieved from non-volatile memory, the anomaly team determined that Spirit experienced only three reboots on the first sol. The repeating reset condition started early the next sol, when the rover woke up for the UHF window.

Within a week, the anomaly team determined that the problem was a design error in the DOS file system library code. The file system mechanism uses a representation of the file system structure (i.e., a Table of Contents or TOC) in RAM, to optimize performance. This TOC is essentially an array with the file name, attributes, date, time, file size, and a pointer to the starting point of the file/directory in the FLASH memory.

When a file is deleted from the file system, the TOC is changed to reflect that the file has been deleted, but the size of the TOC does not shrink. Even though the data management team rigorously managed and deleted files, the TOC retained the knowledge of all the files that *ever* existed in FLASH file system. So after collecting telemetry for seven months of cruise, one day of EDL, one

week of deployments, standup and egress, and one week of driving and science, the TOC grew to consume all the available RAM.

This growth was not supposed to occur. Configuration errors in the flight software allowed the TOC to expand unbounded. When the out-of-memory condition occurred, a critical task silently suspended (the silence was another error in the code). With this critical task suspended, many functions that access the file system were blocked from working correctly. In particular, the shutdown algorithm could not power off the CPU, even though the "delayed reset" allowed enough time.

Each time the out-of-memory event occurred, the system would reset (after the delayed reset time). Many of the on-board communication windows and the operation team's commanded windows were interrupted by these resets. The reason a reset or the use of a different flight software image wouldn't clear the problem is that the TOC is recreated during initialization when the FLASH memory is mounted, causing another out-of-memory event. Crippled mode stopped the resets because the FLASH file system is not used in this mode, and this TOC is not created.

Two weeks after the first symptoms were observed, the anomaly team reformatted the FLASH memory (which also deleted the TOC), so upon initialization, the size of the recreated TOC was small again. Spirit was temporarily cured. For the next two months, the problem was avoided by careful management of the total number of files allowed in the FLASH file system. A new flight software load eventually fixed the deleted-file-representation problem in the TOC, as well as other vulnerabilities in the system.[5]

3.4 Opportunity's Failed-on Heater

On Opportunity's first evening on Mars (the same day Spirit was put into crippled mode), the power team observed an unexpected 0.5 amp current, which appeared around 23:00 LST and disappeared around 09:30 LST. The battery state-of-charge was also lower than predicted. The anomaly team investigated and discovered a failed-on (closed) heater switch for an IDD joint. The external thermostat box cuts off the heater circuit when the external temperatures warm up, but allows the heater to turn on when the temperature cools down (per design). Multiple attempts to open the switch failed. The effect was an extra 180 W-hrs of precious energy consumed every night.

The corrective action was to implement a flight software modification that would purposefully remove the batteries from the power bus at night and power off all of the devices, including the BCB and survival heaters. The algorithm is called "deep sleep" and, when enabled, it autonomously boots up the vehicle at 18:30 LST each night to pull the batteries offline. The reason the algorithm does not remove the batteries from the power bus whenever *any* shutdown occurs, is that the BCB hardware fault protection will put the batteries back online when it detects the bus voltage dropping as the sun sets. So the deep sleep algorithm has to wait for the sun to set low enough that no current is available from the solar arrays. The operations team can temporarily disable deep sleep mode if an early morning UHF communication window is scheduled.

[5] See Ref. [3] for more details on this anomaly.

4 Conclusion

The MER surface fault protection design is incorporated throughout the flight system, in both hardware and software. Autonomously, the rovers maintain a safe thermal and energy balance, as well as communicate to orbiting assets or directly to Earth. Throughout cruise, landing, and over one year of surface operations, the overall system has met many challenges and proven itself in practice on Mars.

5 Acknowledgements

The entire MER design team deserves praise for producing a robust rover design. For making the MER fault protection design truly work as a system, a few core individuals deserve recognition: Adrian Adamson, James Donaldson, Glenn Reeves, Joseph Snyder, and Jason Willis.

The rover design described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

6 References

[1] Fault Protection for JPL Deep Space Missions, G. Mark Brown, Planning Session for MSL Rover Autonomy Validation Workshop, Pasadena, CA, March 3, 2003.

[2] Mars Exploration Rover: Thermal Design is a System Engineering Activity, G. Tsuyuki, A. Avila, H. Awaya, R. Krylo, K. Novak, C. Phillips, 2004-01-2411 2004© SAE International.

[3] The Mars Rover Spirit FLASH Anomaly, G. Reeves, T. Neilson, 0-7803-8870-4/05© 2005 IEEE paper #1426.